(19) JAPANESE PATENT OFFICE

PATENT ABSTRACTS OF JAPAN

(11) Publication number: **09251379 A**

(43) Date of publication of application: **22 . 09 . 97**

(51) Int. Cl **G06F 9/06**
G06F 3/14
G06F 9/46

(21) Application number: **08061464**

(22) Date of filing: **18 . 03 . 96**

(71) Applicant: **TOSHIBA CORP TOSHIBA SOFTWARE ENG KK**

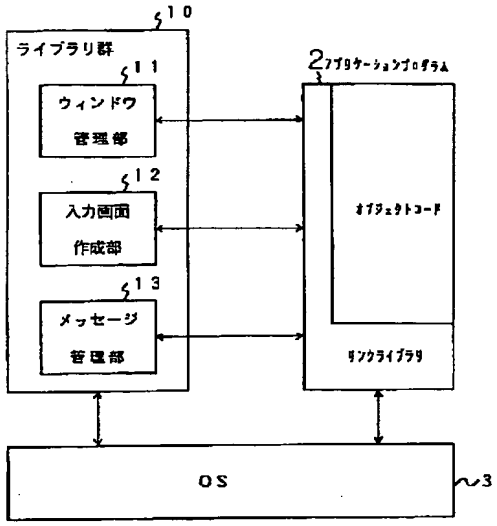
(72) Inventor: **SHIMIZU NOBUO**
KONDO YUJI

(54) **COMPUTER SYSTEM**

(57) Abstract:

PROBLEM TO BE SOLVED: To easily transplant an application program not coping with a window system to the window system.

SOLUTION: The application program 2 not coping with the window system is linked with a library group 10 for transplanting and a window management part 11 prepares a window when the application program 2 is activated. Also, when the application program 2 requests the display of an input screen, an input screen preparation part 12 plots the input screen inside the window and a message management part 13 activates a message loop for receiving a message from an OS 3. Then, when data input from a user is present, by delivering the input data to the application program 2 and ending the message loop, control is returned to the application program 2.



(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号

特開平9－251379

(43)公開日　平成９年(1997)９月22日

(51)Int.Cl.[6]		識別記号	庁内整理番号	ＦＩ		技術表示箇所
Ｇ０６Ｆ	9/06	５４０		Ｇ０６Ｆ	9/06	５４０Ｄ
	3/14	３１０			3/14	３１０Ｅ
	9/46	３４０			9/46	３４０Ｂ

審査請求　未請求　請求項の数２　ＯＬ　（全 12 頁）

(21)出願番号　特願平８－61464

(22)出願日　平成８年(1996)３月18日

(71)出願人　000003078
株式会社東芝
神奈川県川崎市幸区堀川町72番地

(71)出願人　000221133
東芝ソフトウェアエンジニアリング株式会社
東京都青梅市新町1385番地

(72)発明者　清水　伸夫
東京都青梅市末広町２丁目９番地　株式会社東芝青梅工場内

(72)発明者　近藤　祐史
東京都青梅市新町1385番地　東芝ソフトウェアエンジニアリング株式会社内

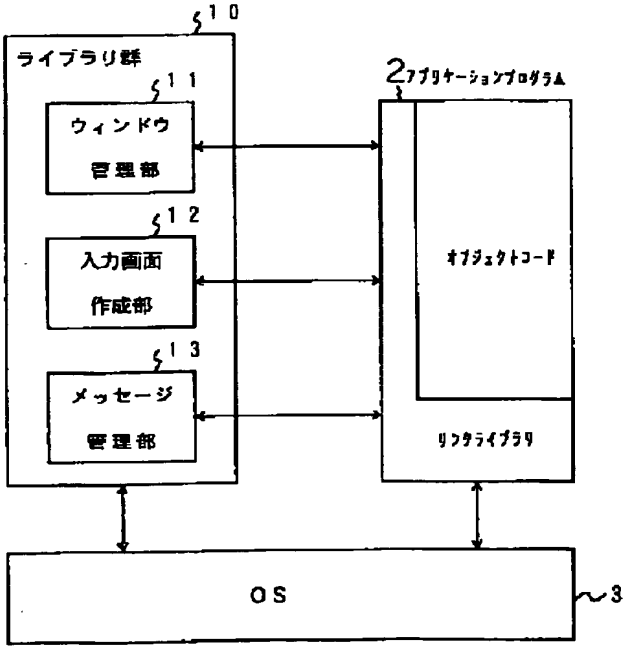
(74)代理人　弁理士　鈴江　武彦

(54)【発明の名称】　コンピュータシステム

(57)【要約】

【課題】ウインドウシステムに対応していないアプリケーションプログラムをウインドウシステムに容易に移植可能とするコンピュータシステムを提供する。

【解決手段】ウインドウシステムに未対応のアプリケーションプログラム２を移植用のライブラリ群１０とリンクづけし、アプリケーションプログラム２の起動時に、ウインドウ管理部１１がウインドウを作成する。また、アプリケーションプログラム２が入力画面の表示を要求したときには、入力画面作成部１２がその入力画面をそのウインドウ内に描画するとともに、ＯＳ３からのメッセージを受信するためのメッセージループをメッセージ管理部１３が起動する。そして、ユーザからのデータ入力があったときに、その入力データをアプリケーションプログラム２に引き渡すとともに、メッセージループを終了させることによって制御をアプリケーションプログラム２に返却する。



1

【特許請求の範囲】

【請求項１】　同一画面上に少なくとも一つ以上のウインドウを表示し、それぞれのウインドウで別々のアプリケーションプログラムを実行可能なオペレーティングシステムであって、前記ウインドウ内で動作中のアプリケーションプログラムに各種メッセージを送信する機能を有するオペレーティングシステムを適用してなるコンピュータシステムにおいて、

前記メッセージの受信機能を含む前記ウインドウ内で動作するための対応がなされていない非対応アプリケーションプログラムと関係づけられ、

前記非対応アプリケーションプログラムの起動時に、前記オペレーティングシステムにウインドウの作成を要求するウインドウ作成手段と、

前記非対応アプリケーションプログラムが入力画面の表示を要求したときに、その入力画面を前記ウインドウ作成手段により作成されたウインドウ内に描画する入力画面作成手段と、

前記入力画面作成手段による前記ウインドウ内への前記入力画面の描画と同期して、前記オペレーティングシステムからのメッセージを受信するためのメッセージループを起動するメッセージループ起動手段と、

前記メッセージループ起動手段により受信したメッセージが、前記非対応アプリケーションプログラムが表示要求した入力画面に対応する入力データを含むものであったときに、その入力データを前記非対応アプリケーションプログラムに送信するとともに前記メッセージループを終了させるメッセージ処理手段と、

前記非対応アプリケーションプログラムの終了時に、前記オペレーティングシステムにウインドウの解放を要求するウインドウ解放手段とを有してなるライブラリ群を具備し、

前記ウインドウ対応がなされていない非対応アプリケーションプログラムを前記ライブラリ群と関係づけるのみで前記ウインドウ環境下に移植可能とすることを特徴とするコンピュータシステム。

【請求項２】　前記メッセージ処理手段は、前記受信したメッセージが前記ウインドウの移動やサイズ変更を含む前記ウインドウ制御のためのメッセージであったときに、前記非対応アプリケーションプログラムに代替してその処理を実行することを特徴とする請求項１記載のコンピュータシステム。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】この発明は、ディスプレイを複数のウインドウで区切り、それぞれのウインドウで別々のアプリケーションプログラムを実行するオペレーティングシステムを適用してなるコンピュータシステムに係り、特にこのウインドウ内で動作するための対応がなされていない非対応アプリケーションプログラムをウインドウシステムに容易に移植することを可能とするコンピュータシステムに関する。

2

【０００２】

【従来の技術】近年、デスクトップをはじめとして、ラップトップ、ノートブックおよびパームトップなど、種々のパーソナルコンピュータが開発されており、その普及は目覚ましいものがある。そして、これらの普及に伴なって、コンピュータシステムとユーザとのインタフェースに様々な工夫がなされてきている。

【０００３】従来では、ＣＵＩ（Ｃｈａｒａｃｔｅｒ　Ｕｓｅｒ　Ｉｎｔｅｒｆａｃｅ）などと称されるインタフェースが主流であり、システムで実行中のアプリケーションプログラムがユーザにデータの入力を促す入力画面を表示し、一方で、ユーザがキーボードを介してコマントを入力することによってインタフェースを確保するといったものであった。

【０００４】しかしながら、このような方法によると、たとえば、ユーザ自身がアプリケーションプログラムと対話するためのコマンドを覚えておかなければならないなどといった問題があり、あまり使い勝手がよいとはいえなかった。

【０００５】そして、このような問題を解決するものとして、ＧＵＩ（ＧｒａｐｈｉｃａｌＵｓｅｒ　Ｉｎｔｅｒｆａｃｅ）などと称されるインタフェースが登場してきた。このＧＵＩは、画面上に絵やアイコンなどを表示して、直感的に分かりやすく操作できるようにしたものであり、マウスなどのポインティングデバイスによって、表示されたアイコンを指定することによって操作を進めていくことが可能である、そして、最近では、このＧＵＩが、コンピュータシステムとユーザとのインタフェースの主流となってきている。

【０００６】ウインドウシステムとは、ディスプレイの画面をいくつかの窓（ウインドウ）に分け、それぞれのウインドウで別々のアプリケーションを動作させることのできるシステムをいう。このウインドウには、画面を上下左右に分割し、その一つ一つをウインドウとして各アプリケーションプログラムに割り当てるものや、ウインドウの重なりを許可して複数のウインドウを作成し、これら複数のウインドウそれぞれに各アプリケーションプログラムを割り当てるなどといったものが存在する。

【０００７】一方、このウインドウシステムを実現するＯＳ配下でアプリケーションプログラムを動作させるためには、アプリケーションプログラム自身もウインドウ対応がなされていなければならない。

【０００８】ウインドウシステムでは、前述したようにＧＵＩが採用されており、これによってユーザは、マウスなどのポインティングデバイスによってほとんどの操作を行なう。そして、このウインドウシステムでは、ウインドウの移動やサイズ変更などもポインティングデバイスで行なえるようになっており、ＯＳは、入力データ

をはじめ、これらの情報をも、そのウインドウ内で動作中のアプリケーションプログラムにメッセージとして通知する仕組みとなっている。

【０００９】すなわち、このような環境下で動作するアプリケーションプログラムは、ＯＳからのメッセージを取得するメッセージ受信機能を備え、かつこのメッセージに対応したメッセージ処理機能を備えていなければならなかった。

【００１０】しかしながら、ウインドウシステムに対応していないアプリケーションプログラムを資源として活用するユーザは依然として多く、また、このようなアプリケーションプログラムを前述したようなウインドウシイステムの環境下に移植するためには、ソースプログラムレベルでの修正が必要であり、オブジェクト提供のアプリケーションプログラムの場合、エンドユーザでは移植が不可能であるといった問題があった。

【００１１】

【発明が解決しようとする課題】このように、最近のパーソナルコンピュータに適用されるＯＳは、ウインドウシステムを前提としているものがほとんどであるが、このようなウインドウシステムに対応していないアプリケーションプログラムを資源として活用するユーザは依然として多く、また、このようなアプリケーションプログラムを前述したようなウインドウシイステムの環境下に移植するためには、ソースプログラムレベルでの修正が必要であり、オブジェクト提供のアプリケーションプログラムの場合、エンドユーザでは移植が不可能であるといった問題があった。

【００１２】この発明はこのような実情に鑑みてなされたものであり、ウインドウシステムで動作するための対応がなされていない非対応アプリケーションプログラムを容易にウインドウシステムに移植することを可能とするコンピュータシステムを提供することを目的とする。

【００１３】

【課題を解決するための手段】この発明は、同一画面上に少なくとも一つ以上のウインドウを表示し、それぞれのウインドウで別々のアプリケーションプログラムを実行可能なオペレーティングシステムであって、前記ウインドウ内で動作中のアプリケーションプログラムに各種メッセージを送信する機能を有するオペレーティングシステムを適用してなるコンピュータシステムにおいて、前記メッセージの受信機能を含む前記ウインドウ内で動作するための対応がなされていない非対応アプリケーションプログラムと関係づけられ、前記非対応アプリケーションプログラムの起動時に、前記オペレーティングシステムにウインドウの作成を要求するウインドウ作成手段と、前記非対応アプリケーションプログラムが入力画面の表示を要求したときに、その入力画面を前記ウインドウ作成手段により作成されたウインドウ内に描画する入力画面作成手段と、前記入力画面作成手段による前記

ウインドウ内への前記入力画面と同期して、前記オペレーティングシステムからのメッセージを受信するためのメッセージループを起動するメッセージループ起動手段と、前記メッセージループ起動手段により受信したメッセージが、前記非対応アプリケーションプログラムが表示要求した入力画面に対応する入力データを含むものであったときに、その入力データを前記非対応アプリケーションプログラムに送信するとともに前記メッセージループを終了させるメッセージ処理手段と、前記非対応アプリケーションプログラムの終了時に、前記オペレーティングシステムにウインドウの解放を要求するウインドウ解放手段とを有してなるライブラリ群を具備し、前記ウインドウ対応がなされていない非対応アプリケーションプログラムを前記ライブラリ群と関係づけるのみで前記ウインドウ環境下に移植可能とすることを特徴とする。

【００１４】この発明によれば、ウインドウシステム環境下にウインドウ内で動作するための対応がなされていない非対応アプリケーションプログラムを移植する際、ユーザは、そのアプリケーションプログラムと本発明の移植用のライブラリ群とをリンクする。

【００１５】たとえば、アプリケーションプログラムがＤＬＬ（Ｄｙｎａｍｉｃ　ＬｉｎｋＬｉｂｒａｒｙ）を使用する場合、そのアプリケーションプログラムがどのＤＬＬを使用するのかを決定するために、アプリケーションプログラムの作成時にインポートライブラリをリンクする。そして、ＯＳは、このインポートライブラリのリンクにより、そのアプリケーションプログラムがどのＤＬＬを利用して動作するものなのかを認識することができる。したがって、この場合、非対応アプリケーションプログラムは、このインポートライブラリのリンクによって、移植用のライブラリ群（ＤＬＬ）と関係づけられることになる。

【００１６】従来のアプリケーションプログラムでは、画面に入力画面を表示して、この画面に対してユーザが入力した情報を入力データとして取り込み、この取り込んだ入力データに基づいて処理を進行させていくといった逐次的な動作が行なわれていく。

【００１７】そこで、この発明では、移植用のライブラリ群が、以下のタイミングで以下のウインドウ処理を代替することによって、非対応アプリケーションプログラムをウインドウシステムに移植する。

（１）起動時

ＯＳへのウインドウの作成の要求。

（２）入力画面の表示時

ウインドウ内での入力画面の描画。

【００１８】ＯＳからのメッセージを受信するためのメッセージループの起動

（３）ＯＳからのメッセージ受信時。

ａ）入力データの場合

アプリケーションプログラムへの入力データの転送。

【００１９】メッセージループの終了（これにより、アプリケーションプログラムに制御が返却される）。

ｂ）移動などのウインドウ制御の場合

メッセージに応じたウインドウ処理。

（４）終了時

ＯＳへのウインドウの解放の要求。

【００２０】これにより、ＯＳからはウインドウシステム対応のアプリケーションプログラムとして認識され、かつアプリケーションプログラムは従前のインタフェースで入力データを取り込みながら処理を行なえることになり、さらにアプリケーションプログラムの修正などを一切必要としない。

【００２１】

【発明の実施の形態】以下、図面を参照してこの発明の一実施形態を説明する。図１は、本実施形態のコンピュータシステムの概略構成を示すブロック図である。

【００２２】図１に示すように、本実施形態のコンピュータシステムは、ＯＳ（オペレーティングシステム）３によって全体が制御されている。また、このＯＳ３配下で動作するアプリケーションプログラム２は、ＤＬＬ（Ｄｙｎａｍｉｃ　ＬｉｎｋＬｉｂｒａｒｙ）１に格納された複数の機能オブジェクト（ライブラリ群１０ａ～１０ｂ）のうち、必要とする機能オブジェクトと関連づけられて動作する。そして、本実施形態の特徴とする移植用のライブラリ群も、このＤＬＬ１に格納され、ウインドウシステム非対応のアプリケーションプログラムは、この移植用のライブラリ群と関連づけられることになる。

【００２３】なお、これらアプリケーションプログラム２と、ＤＬＬ１中の意図するライブラリ群との関係づけは、アプリケーションプログラム２の作成時に、インポートライブラリ（リンクライブラリ）をリンクすることによって実現される。そして、ＯＳ３は、このインポートライブラリのリンクにより、アプリケーションプログラム２それぞれが、どのライブラリ群能を利用して動作するものなのかを認識する。

【００２４】図２には、本実施形態の移植用のライブラリ群の概略構成が示されている。図２に示すように、本実施形態の移植用ライブラリ群１０は、ウインドウ管理部１１、入力画面作成部１２およびメッセージ管理部１３を具備してなる。

【００２５】ウインドウ管理部１１は、非対応アプリケーションプログラム２の稼働時に、ＯＳ３に対してウインドウの作成を要求するとともに、非対応アプリケーションプログラム２の終了時に、ＯＳ３に対してウインドウの解放を要求する。

【００２６】入力画面作成部１２は、非対応アプリケーションプログラム２が入力画面の表示を要求したときに、ウインドウ管理部１１によって作成されたウインドウ内にその入力画面を描画する。

【００２７】そして、メッセージ管理部１３は、非対応アプリケーションプログラム２による入力画面の表示要求に基づいて入力画面作成部１２が入力画面を描画したときに、ＯＳ３からのメッセージを受信するためのメッセージループを起動し、たとえばウインドウの移動やサイズ変更などといったメッセージの種別に応じた処理を行なうとともに、入力データの非対応アプリケーションプログラム２への引き渡しおよびメッセージループの終了を行なう。

【００２８】ここで、図３を参照してウインドウシステムへの対応がなされていないアプリケーションプログラムの従来の環境での動作手順を説明しておく。まず、非対応アプリケーションプログラムは、ＯＳに対して入力画面の作成と表示を要求する（ステップＦ１）。ユーザがこの入力画面に対応したデータ入力を行なうと、そのデータはＯＳから非対応アプリケーションプログラムに引き渡され、非対応アプリケーションプログラムは、この引き渡された入力データに基づいた処理を実行する（ステップＦ２）。

【００２９】すなわち、非対応アプリケーションプログラムは、ステップＦ１～ステップＦ２を必要なデータ分だけ繰り返し、逐次的な処理を進行させていく。図４には、ウインドウシステムへの対応がなされていないアプリケーションプログラムの従来の環境での動作原理が示されている。

【００３０】非対応アプリケーションプログラムが、ＯＳに対して入力画面の作成と表示を要求した後（図４の（１））、ＯＳからユーザの入力したデータが送信されると（図４の（２））、非対応アプリケーションプログラムは、この送信された入力データに基づいた処理を実行する（図４の（３））。

【００３１】同様に、次のデータが必要な場合には、ＯＳに対して入力画面の作成と表示を要求し（図４の（４））、この要求に対して返送された入力データ（図４の（５））に基づいて処理を実行する。

【００３２】このように、従来の非対応アプリケーションプログラムは、必要なデータを受け取りつつ、逐次的な処理を進行させていく。次に、図５を参照してウインドウシステムへの対応がなされたアプリケーションプログラムの動作手順を説明する。

【００３３】まず、対応アプリケーションプログラムは、ＯＳからのメッセージを入力するためのメッセージループを起動する（ステップＧ１）。このメッセージループには、メッセージの受信処理と、そのメッセージの種別に応じたディスパッチ処理とが含まれている。

【００３４】このメッセージループ起動後、対応アプリケーションプログラムは、ＯＳからのメッセージを待機し（ステップＧ２）、メッセージを受信したときに（ステップＧ３のＹ）、このメッセージが終了を指示するメ

ッセージかどうか判定し（ステップＧ４）、終了を指示するメッセージでなければ（ステップＧ４のＮ）、そのメッセージの種別に基づく処理を実行する（ステップＧ５）。

【００３５】したがって、このとき、ウインドウの移動やサイズ変更などのウインドウ制御に関するメッセージを受信したときには、その処理を行なうモジュールをディスパッチし、ユーザなどからの入力データを含むメッセージを受信したときには、そのデータの処理を行なうモジュールをディスパッチする。

【００３６】図６には、ウインドウシステムへの対応がなされたアプリケーションプログラムの動作原理が示されている。対応アプリケーションプログラムが、ＯＳからのメッセージを入力するためのメッセージループを起動すると（図６の（１））、これ以降、ユーザからのデータ入力をはじめ、ウインドウの移動やサイズ変更など、イベントが発生するたびにＯＳからメッセージが送信され、その都度、そのメッセージに基づいた処理が行なわれる（図６の（２）～（３），（４）～（５），（６）～）。そして、この処理は、対応アプリケーションプログラムが終了するまで繰り返される。

【００３７】ここで、本実施形態の環境（ウインドウシステム環境）におけるウインドウシステムへの対応がなされていないアプリケーションプログラムの動作手順を説明する。

【００３８】本実施形態のコンピュータシステムでは、図２に示したように、非対応アプリケーションプログラム２を移植用のライブラリ群１０とリンクさせることによって、実行プログラムを作成する。以下、この実行アプリケーションの動作手順を説明する。

【００３９】図７乃至図１１には移植用のライブラリ群１０とリンクされた非対応アプリケーションプログラム２との動作手順が示されている。非対応アプリケーションプログラム２は、起動すると、まず、初期処理を実行する（図７のステップＡ１）。このとき、非対応アプリケーションプログラム２では、「Ｏｐｅｎ」などといった、起動時に必須の関数を発行することになり、この関数の発行は、ＯＳ３の制御によってリンクづけられた移植用のライブラリ群１０に通知されることになる。

【００４０】この通知を受けたときの移植用のライブラリ群１０の動作を図８に示す。この通知を受け取った移植用のライブラリ群１０では、ウインドウ管理部１１が、非対応アプリケーションプログラム２がウインドウシステム環境で動作するための各種資源を確保する（図８のステップＢ１）。そして、ウインドウ管理部１１は、非対応アプリケーションプログラム２が動作するためのウインドウの作成を非対応アプリケーションプログラム２に変わって実行する（図８のステップＢ２）。

【００４１】この後、非対応アプリケーションプログラム２が、データを入力するための画面表示を要求する関数を発行すると（図７のステップＡ２）、同様にこの関数の発行は、ＯＳ３の制御によってリンクづけられた移植用のライブラリ群１０に通知され、この通知を受け取った移植用のライブラリ群１０では、入力画面作成部１２が、図９に示すような動作を行なう。

【００４２】すなわち、ＯＳ３に対してウインドウプロシージャの呼び出しを実行し（図９のステップＣ１）、非対応アプリケーションプログラム２から受け取った表示要求に基づいたウインドウプロシージャ内での描画処理を実行する（図９のステップＣ２）。

【００４３】そして、この表示した画面を介したデータの入力が非対応アプリケーションプログラム２から要求されると、移植用のライブラリ群１０では、メッセージ管理部１３が、図１０に示す処理を実行する。

【００４４】まず、メッセージ管理部１３は、ＯＳ３がウインドウ管理部１１によって作成されたウインドウに対して送信するメッセージを受信するためのメッセージループを起動する（図１０のステップＤ１）。これにより、入力画面作成部１２によって描画された表示画面に対応してユーザが入力したデータを含むメッセージを受信可能になるとともに、ウインドウの移動やサイズ変更などを指示するメッセージも受信可能となる。

【００４５】そして、メッセージ管理部１３は、受信されたメッセージに応じてディスパッチを実行し、その処理を行なう（図１０のステップＤ２）。なお、ウインドウプロシージャでのキー入力の確認をした結果、ユーザからの入力データであった場合には、非対応アプリケーションプログラム２が参照可能な内部キーバッファに入力されたキーコードを登録し（図１０のステップＤ３）、メッセージループを終了させる（図１０のステップＤ４）。すなわち、このメッセージループの終了によって、非対応アプリケーションプログラム２側に制御が返却され、非対応アプリケーションプログラム２側での実行が再開されることになる。

【００４６】そして、非対応アプリケーションプログラム２が終了処理を実行したときに（図７のステップＡ４）、移植用のライブラリ群１０では、ウインドウ管理部１１が、図１１に示すような手順で処理を行なう。

【００４７】ウインドウ管理部１１は、非対応アプリケーションプログラム２の終了時に確保した各種資源を解放し（図１１のステップＥ１）、ＯＳ３に対してウインドウの破壊を要求する（図１１のステップＥ２）。

【００４８】これにより、非対応アプリケーションプログラム２は、ウインドウシステムの環境を意識せずに動作することができる結果、ソースレベルでの修正などが一切必要なくなり、かつＯＳ３からはウインドウシステム対応のアプリケーションプログラムとして認識されることになる。

【００４９】このときの動作原理を図１２を参照して説明する。アプリケーションプログラム２が起動され、初

期化処理が実行されると（図１２の（１））、移植用ライブラリ群１０のウインドウ管理部１１は、ＯＳ３に対してアプリケーションプログラム２用のウインドウの作成などを要求する（図１２の（２））。ここでのウインドウ管理部１１の処理終了がアプリケーションプログラム２に通知されると（図１２の（３））、次に、アプリケーションプログラム２は、データ入力用画面作成要求を移植用ライブラリ群に通知する（図１２の（４））。この通知を受けた移植用ライブラリ群１０では、入力画面作成部１２が、先に作成したウインドウのどの位置に文字や罫線を描画すべきかを判断してそのデータ入力用画面を作成されたウインドウ内に描画する（図１２の（５））。このとき、入力画面作成部１２は、文字を書く要求などを状況に応じて文字単位や文字列単位でＯＳ３に発行する。そして、入力画面作成部１２は、描画要求のすべてが満たされると、アプリケーションプログラム２に対して処理を返却する（図１２の（６））。

【００５０】アプリケーションプログラム２は、その処理中にデータの入力が必要になると、移植用ライブラリ群１０に対してデータ入力要求を発行する（図１２の（７））。この要求を受け付けた移植用ライブラリ群１０では、メッセージ管理部１３が、ＯＳ３からメッセージを受信するためのメッセージループを起動する（図１２の（８））。

【００５１】なお、このとき、ウインドウシステムでは、要求したデータ入力以外のメッセージ、たとえばウインドウの移動やサイズの変更などを示すメッセージを受信することも考えられる（図１２の（９））。この場合、移植用ライブラリ群１０では、この受信メッセージの要求を満たす処理を実行し（図１２の（１０））、さらに期待するデータの入力を待機する。

【００５２】また、その期待するデータを含むメッセージが受信された場合（図１２の（１１））、メッセージ管理部１３は、メッセージループを終了させるとともに（図１２の（１２））、このメッセージに含まれる入力データをアプリケーションプログラム２に引き渡す（図１２の（１３））。これにより、アプリケーションプログラム２に制御が返却される。

【００５３】そして、アプリケーションプログラム２は、この受け取った入力データに基づく処理を実行する。これ以降、たとえばユーザからのデータ入力を必要とする場合には、再度同様の処理を繰り返す。そして、アプリケーションプログラム２が終了処理を実行したときに、移植用ライブラリ群１０のウインドウ管理部１１は、アプリケーションプログラム２の起動時に確保した各種資源の解放や作成したウインドウの破壊処理を実行する。

【００５４】なお、本実施形態では、メッセージループを終了させることによって非対応アプリケーションプログラム２に制御を返却するため、たとえば制御を非対応アプリケーションプログラム２に返却した後にウインドウの移動が指示された場合には、このメッセージの受信が即座には行なえないことになるが、次に非対応アプリケーションプログラム２からデータの入力要求が行なわれた際に、このウインドウ処理が真っ先に実行されることになるため、問題はないと思われる。

【００５５】これにより、ユーザは、ウインドウシステムに対応していないアプリケーションプログラムを容易かつ安価にウインドウシステムに移植することができることになる。

【００５６】

【発明の効果】以上詳述したように、この発明によれば、ウインドウ対応がなされていない非対応アプリケーションプログラムを移植用のライブラリ群と関係づけるのみでウインドウ環境下に移植することができるため、ソースレベルでの修正などが一切不要となり、かつウインドウシステムに対応させるための知識も必要とせずに容易かつ安価に移植作業を行なうことができる。

【図面の簡単な説明】

【図１】この発明の実施形態のコンピュータシステムの概略構成を示すブロック図。

【図２】同実施形態の移植用のライブラリ群の概略構成を示す図。

【図３】ウインドウシステムへの対応がなされていないアプリケーションプログラムの従来の環境での動作手順を説明するフローチャート。

【図４】ウインドウシステムへの対応がなされていないアプリケーションプログラムの従来の環境での動作原理を示す図。

【図５】ウインドウシステムへの対応がなされたアプリケーションプログラムの動作手順を説明するフローチャート。

【図６】ウインドウシステムへの対応がなされたアプリケーションプログラムの動作原理を示す図。

【図７】同実施形態の環境でのウインドウシステムへの対応がなされていないアプリケーションプログラムの動作手順を説明するフローチャート。

【図８】同実施形態の移植用のライブラリ群の動作手順を説明するフローチャート。

【図９】同実施形態の移植用のライブラリ群の動作手順を説明するフローチャート。

【図１０】同実施形態の移植用のライブラリ群の動作手順を説明するフローチャート。

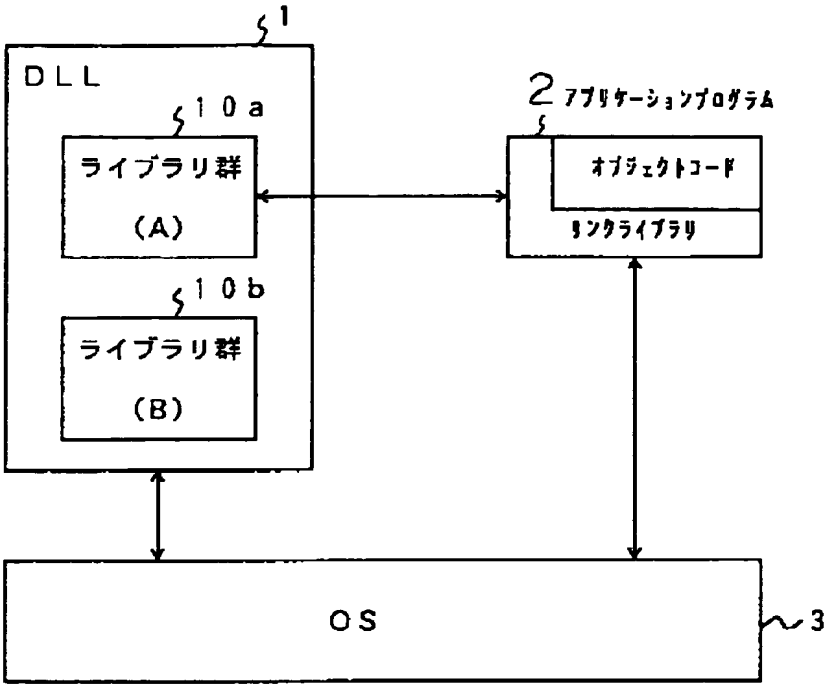
【図１１】同実施形態の移植用のライブラリ群の動作手順を説明するフローチャート。

【図１２】同実施形態の動作原理を説明する図。

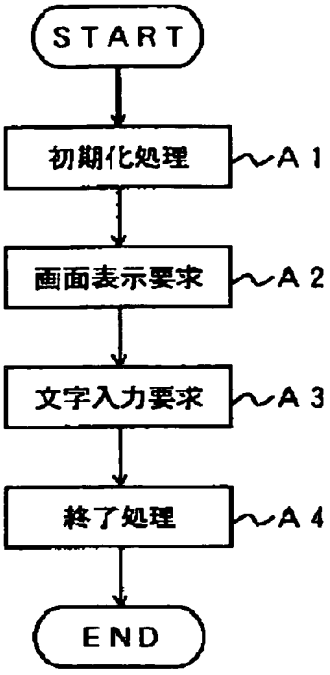
【符号の説明】

１…ＤＬＬ、２…アプリケーションプログラム、３…ＯＳ、１１…ウインドウ管理部、１２…入力画面作成部、１３…メッセージ管理部。

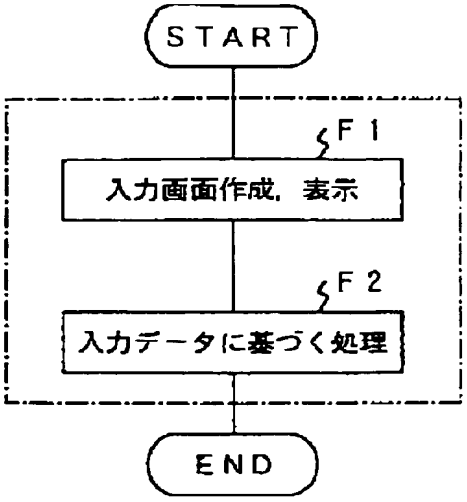
【図１】



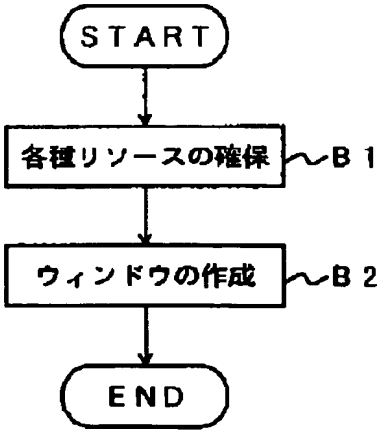
【図７】



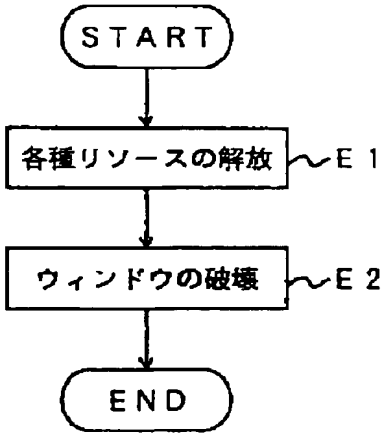
【図３】



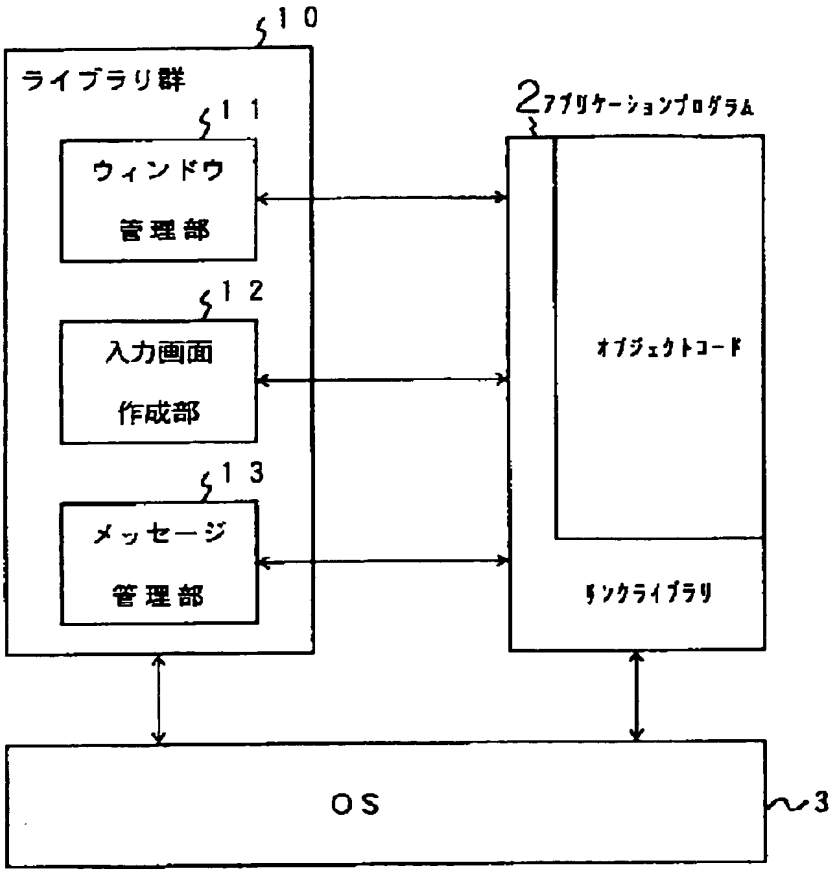
【図８】



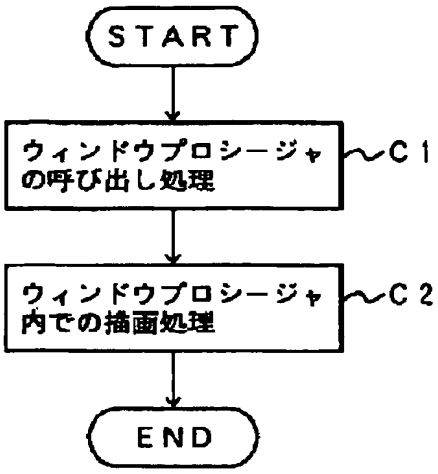
【図１１】



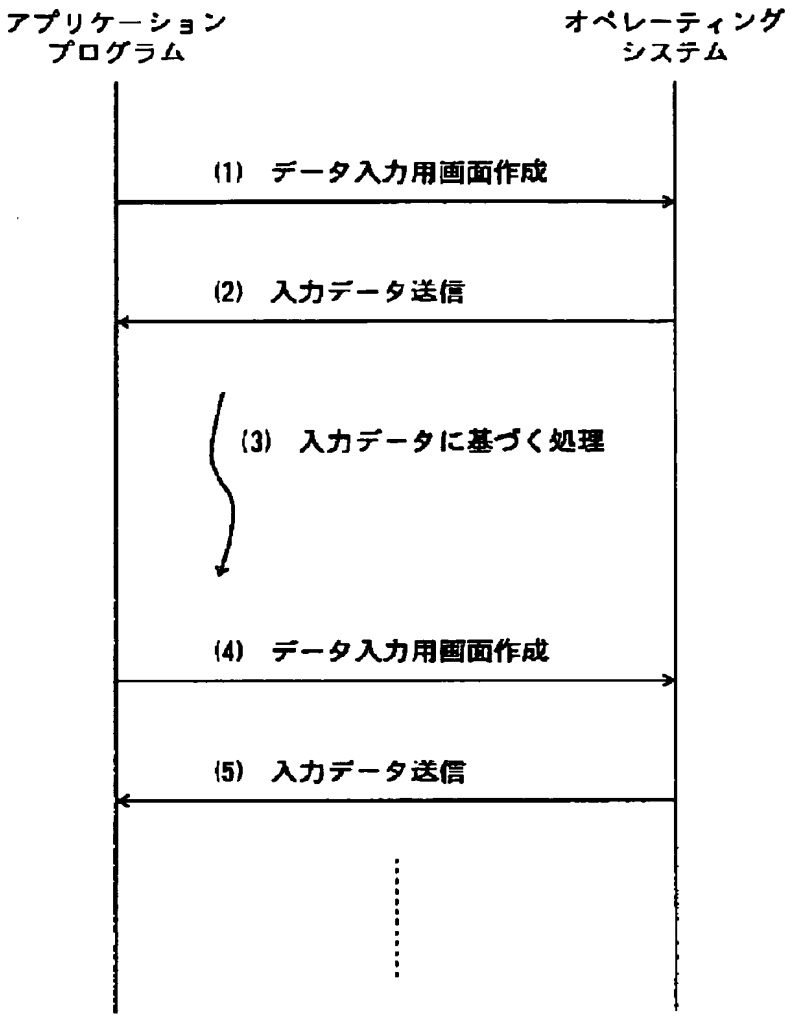
【図２】


10
ライブラリ群
11
ウィンドウ
管理部
12
入力画面
作成部
13
メッセージ
管理部
2
アプリケーションプログラム
オブジェクトコード
リンクライブラリ
OS
3

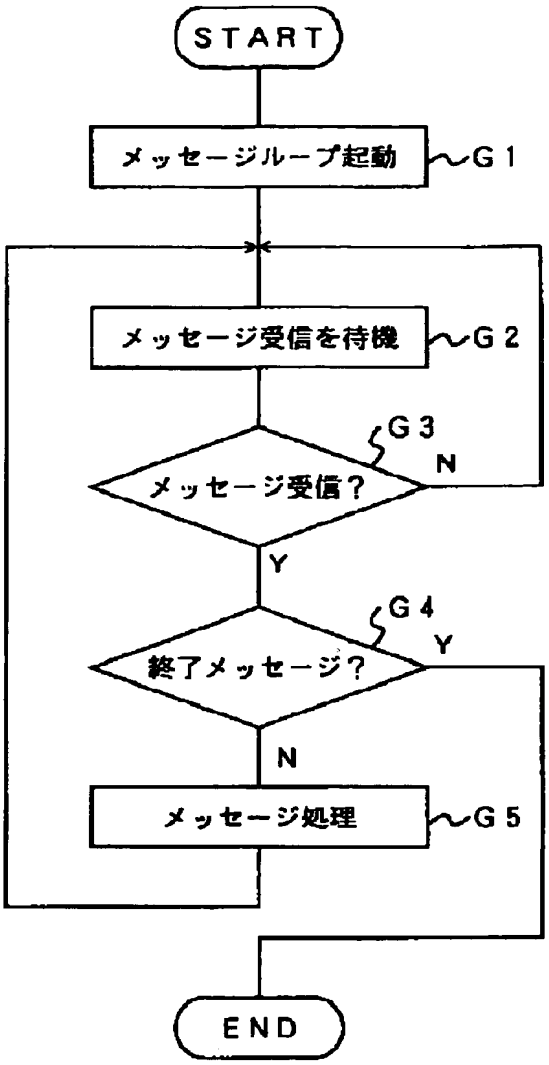
【図９】


START
ウィンドウプロシージャ
の呼び出し処理
C1
ウィンドウプロシージャ
内での描画処理
C2
END

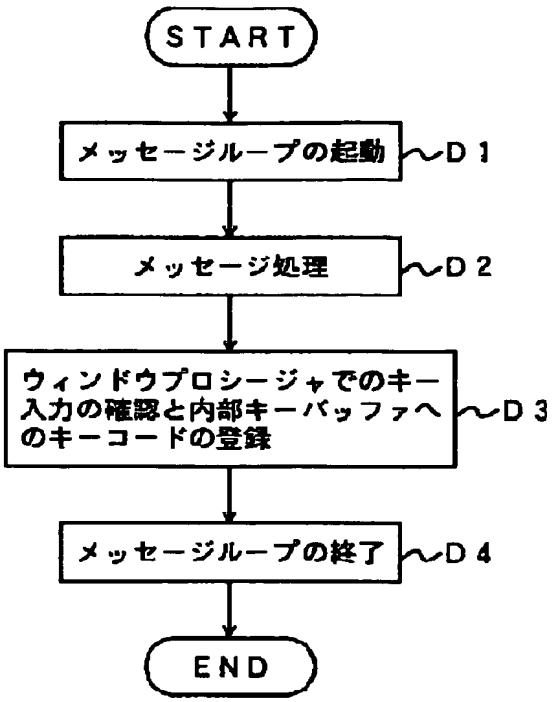
【図４】


アプリケーション
プログラム
オペレーティング
システム
(1)　データ入力用画面作成
(2)　入力データ送信
(3)　入力データに基づく処理
(4)　データ入力用画面作成
(5)　入力データ送信

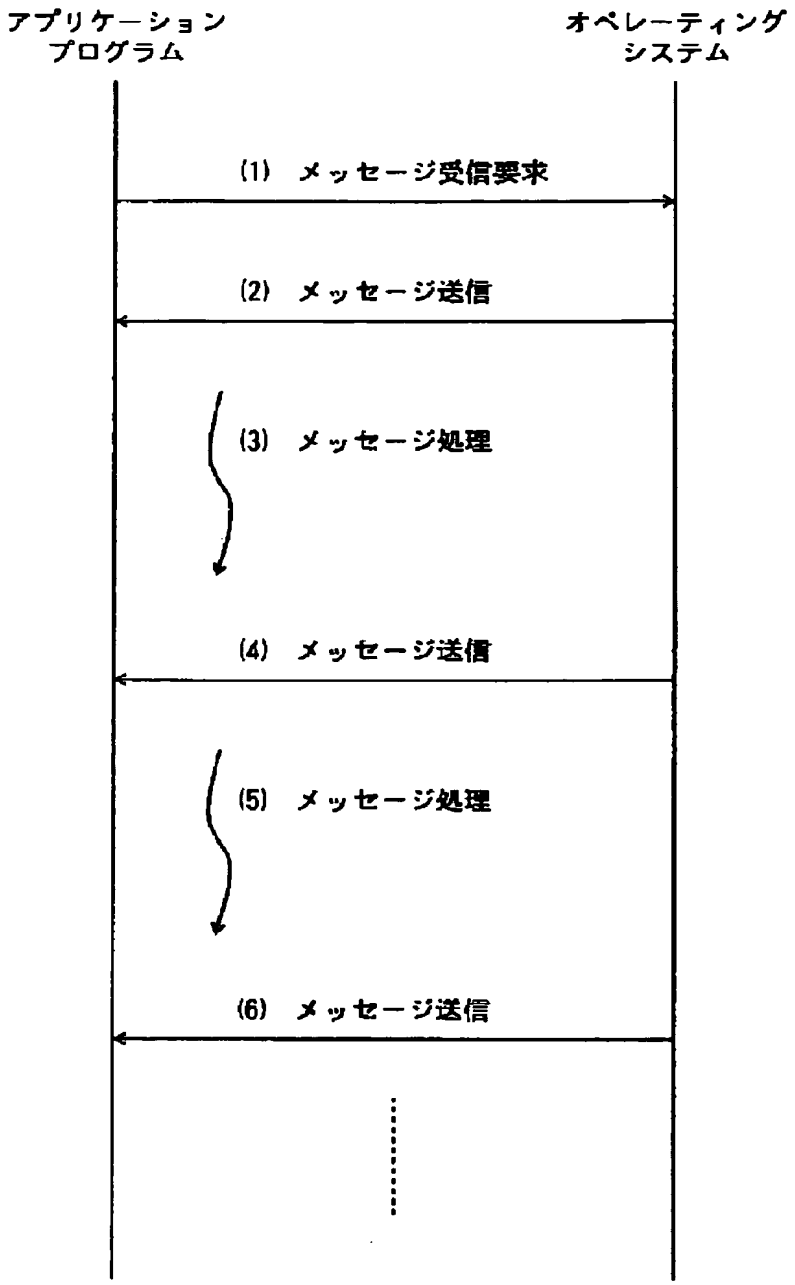
【図５】


START
メッセージループ起動
G 1
メッセージ受信を待機
G 2
G 3
メッセージ受信？
N
Y
G 4
終了メッセージ？
Y
N
メッセージ処理
G 5
END

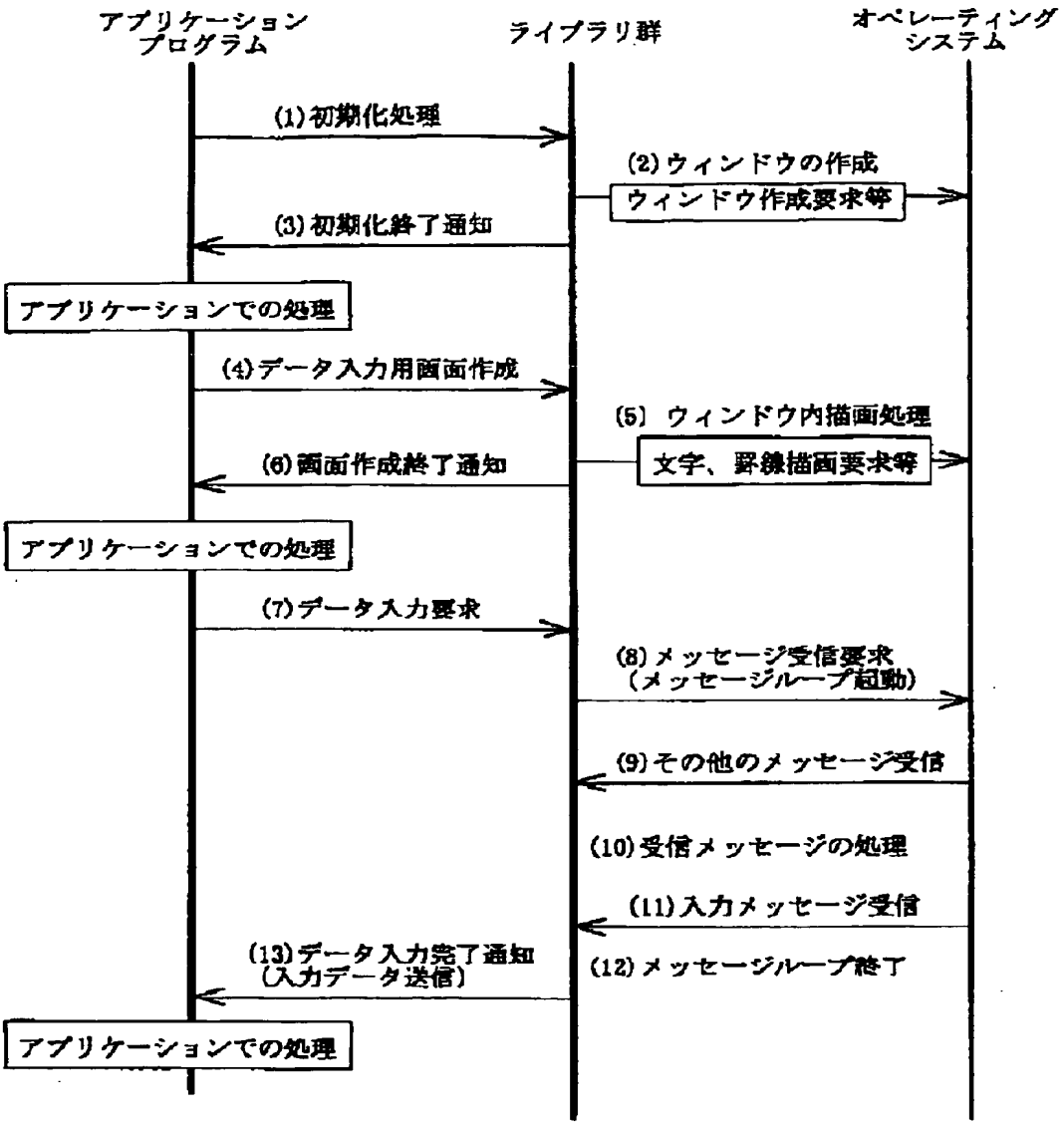
【図１０】


START
メッセージループの起動
D 1
メッセージ処理
D 2
ウィンドウプロシージャでのキー入力の確認と内部キーバッファへのキーコードの登録
D 3
メッセージループの終了
D 4
END

【図６】


アプリケーション
プログラム
オペレーティング
システム
(1) メッセージ受信要求
(2) メッセージ送信
(3) メッセージ処理
(4) メッセージ送信
(5) メッセージ処理
(6) メッセージ送信

【図１２】


アプリケーション
プログラム
ライブラリ群
オペレーティング
システム
(1)初期化処理
(2)ウィンドウの作成
ウィンドウ作成要求等
(3)初期化終了通知
アプリケーションでの処理
(4)データ入力用画面作成
(5) ウィンドウ内描画処理
文字、罫線描画要求等
(6)画面作成終了通知
アプリケーションでの処理
(7)データ入力要求
(8)メッセージ受信要求
(メッセージループ起動)
(9)その他のメッセージ受信
(10)受信メッセージの処理
(11)入力メッセージ受信
(13)データ入力完了通知
(入力データ送信)
(12)メッセージループ終了
アプリケーションでの処理